Maestría en Ingeniería Electrónica

Dirección de Postgrado - FPUNA

Materia: Sistemas de Información Web

Profesores:

Dr. Julio Cesar Mello Roman

Mag. Marcos Benítez

Tarea 2:

Explorando CSS

Alumno:

- Enrique Antonio Flecha Mendoza

Marzo 2025

Contenido

Instrucciones de la tarea

- Preparar un reporte técnico sobre las exploraciones realizadas. Agregar referencias utilizadas al reporte y levantar con los ejercicios desarrollados en un archivo comprimido denominado nombre_apellido_tarea02.zip

Desarrollar las siguientes actividades

1. Investigar técnicas para combinar selectores de CSS y la utilización de herencias.
2. Explicar las prevalencias entre selectores CSS.
3. Investigar sobre framework de CSS
4. Realizar tres ejemplos utilizando animaciones con CSS sin javascript

1. Técnicas para combinar selectores de CSS y la utilización de herencias.

En esta sección se realiza un recorrido por los conceptos de Selectores y Combinadores CSS y luego se explora la utilización de herencias.

1.1. Selectores CSS

Los selectores de atributos en CSS permiten seleccionar elementos que poseen un determinado tipo de atributo o valor. Esto es útil para poder formatear o dar un estilo específico sin recurrir a clases o ID.

1.2. Combinadores CSS

En la utilización de selectores CSS se puede tener más de un simple selector. En estos casos se combina a los mismos con un Combinador CSS. Existen 4 tipos de Combinadores CSS:

- Descendant combinator (space)
- Child combinator (>)
- Next sibling combinator (+)
- Subsequent-sibling combinator (~)

1.2.1. Descendant combinator (space)

Selecciona todos aquellos elementos que son descendientes de un elemento específico. Por ejemplo:

```
div p{
  background-color: yellow;
}
```

selecciona todos los elementos p que se encuentren dentro del elemento div.

1.2.2. Child Combinator (>)

Selecciona todos aquellos elementos que son hijos de un elemento específico. Por ejemplo:

```
div > p {
  background-color: yellow;
}
```

selecciona todos los elementos p que son hijos de div.

1.2.3. Next Sibling Combinator (+)

Es utilizado para seleccionar un elemento que se encuentra inmediatamente a continuación de otro elemento específico. Elementos *Sibling* (en español, hermano) deben tener el mismo elemento padre, además de encontrarse a continuación, como mencionado. Por ejemplo:

```
div + p {
  background-color: yellow;
}
```

selecciona el primer elemento p ubicado inmediatamente posterior al elemento div.

1.2.4. Subsequent-sibling Combinator (~)

Selecciona todos los elementos hermanos subsecuentes a un elemento específico. Por ejemplo:

```
div ~ p {
  background-color: yellow;
}
```

selecciona todos los elementos p que son hermanos subsecuentes de elementos div.

1.3. Herencia

La **herencia** en CSS es el mecanismo por el cual algunos estilos definidos en un elemento se transmiten automáticamente a sus elementos hijos en el árbol del DOM. Esto permite definir propiedades en un contenedor y aplicarlas a múltiples elementos sin necesidad de repetir instrucciones.

Propiedades heredables: Los elementos heredan propiedades relacionadas con el texto y la tipografía, tales como color, font-family, font-size, line-height, y text-align.

Propiedades no heredables: Propiedades relacionadas con el modelo de caja, como margin, padding, border y background, no se heredan. En estos casos, cada elemento debe definir sus propios valores o utilizar técnicas específicas para compartir estilos.

1.3.1. Forzando la Herencia y el Manejo de Valores

Valor inherit**:** Se puede forzar que una propiedad se herede, incluso cuando no es heredable de forma predeterminada:

```
.custom-button {
   border-color: inherit;
}
```

Valores initial **y** unset**:** Estos valores permiten reiniciar o quitar la herencia en ciertos casos.

- initial asigna el valor inicial definido en la especificación de CSS.
- unset actúa como inherit para propiedades heredables y como initial para las no heredables.

```
.reset-style {
   margin: unset;
   color: initial;
}
```

1.3.2. Ventajas de Aprovechar la Herencia

1. **Mantenibilidad:** Permite definir estilos generales en un contenedor (por ejemplo, el <body>) que se extienden a lo largo de toda la aplicación, reduciendo la necesidad de repetir reglas.
2. **Consistencia:** Al utilizar herencia, se consigue un aspecto coherente en toda la aplicación sin tener que redefinir valores para cada componente.
3. **Optimización del Código:** El uso adecuado de la herencia reduce la cantidad de código CSS, lo que facilita la lectura, el mantenimiento y la escalabilidad.

2. Prevalencias entre selectores CSS

La prevalencia (o especificidad) en CSS establece que si dos o más selectores CSS apunta a un mismo elemento, el selector CSS con mayor especificidad es el que será aplicado al elemento en cuestión. También se podría presentar como análogo al concepto de jerarquía, ya que el selector de mayor jerarquía sería el aplicado al elemento en cuestión. En este sentido, la prevalencia, especificidad o jerarquía se establece según se muestra en la siguiente tabla:

Priority	Example	Description
Inline style	<h1 style="color: pink;">	Highest priority, directly applied with the style attribute
Id selectors	#navbar	Second highest priority, identified by the unique id attribute of an element
Classes and pseudo-classes	.test, :hover	Third highest priority, targeted using class names
Attributes	[type="text"]	Low priority, applies to attributes
Elements and pseudo-elements	h1, ::before, ::after	Lowest priority, applies to HTML elements and pseudo-elements

Tabla 1 - Jerarquía de especificidad. Fuente: https://www.w3schools.com/css/css_specificity.asp

3. Framework de CSS

Un framework de CSS es una biblioteca de código predefinido que incluye estilos, componentes y estructuras reutilizables para el diseño de sitios web. Estos frameworks permiten a los desarrolladores implementar rápidamente elementos como cuadrículas, botones, formularios y otros componentes de interfaz de usuario, sin necesidad de escribir CSS desde cero.

3.1. Beneficios:

- **Ahorro de tiempo:** Proporcionan estilos listos para usar, acelerando el desarrollo.
- **Consistencia:** Garantizan un diseño uniforme en todo el proyecto.
- **Responsividad:** Muchos frameworks incluyen sistemas de cuadrículas adaptables para dispositivos móviles.
- **Facilidad de uso:** Reducen la curva de aprendizaje para desarrolladores principiantes.

3.2. Frameworks de CSS Populares

3.2.1. Bootstrap

- **Descripción:** Uno de los frameworks más populares, desarrollado por Twitter. Ofrece un enfoque "mobile-first" y un sistema de cuadrículas flexible.
- **Características:** Componentes predefinidos como botones, modales y menús desplegables.
- **Ventajas:** Amplia documentación y comunidad activa.
- **Ideal para:** Proyectos que requieren rapidez y diseño responsivo.

3.2.2. Tailwind CSS

- **Descripción:** Un framework "utility-first" que permite aplicar clases de utilidad directamente en el HTML.

- **Características:** Altamente personalizable, sin componentes predefinidos.
- **Ventajas:** Flexibilidad para crear diseños únicos.
- **Ideal para:** Desarrolladores que buscan control total sobre el diseño.

3.2.3. Materialize

- **Descripción:** Basado en el diseño "Material Design" de Google, ofrece componentes visuales modernos.
- **Características:** Incluye animaciones y estilos consistentes con las directrices de Google.
- **Ventajas:** Fácil de integrar y con un diseño atractivo.
- **Ideal para:** Aplicaciones que buscan un diseño limpio y profesional.

3.2.4. Foundation

- **Descripción:** Un framework avanzado desarrollado por Zurb, conocido por su flexibilidad.
- **Características:** Sistema de cuadrículas avanzado y herramientas para accesibilidad.
- **Ventajas:** Ideal para proyectos complejos y escalables.
- **Ideal para:** Equipos que necesitan personalización y soporte para accesibilidad.

4. Ejemplos de animaciones con CSS (sin Javascript)

En la misma carpeta comprimida en la que se presenta este informe, se presentan 3 ejemplos de animaciones con CSS (ver archivos en la carpeta comprimida).

5. Bibliografía y fuentes

- HTML: https://www.w3schools.com/html/default.asp
- CSS: https://www.w3schools.com/css/default.asp
- Especificidad https://www.w3schools.com/css/css_specificity.asp
- Animaciones y ejemplos: https://www.w3schools.com/css/css3_animations.asp